7

Computational logic and integer programming

H. Paul Williams and Sally C. Brailsford

Faculty of Mathematical Studies,
University of Southampton, Southampton SO17 1BJ, England

1 Introduction

The enormous increase in the availability and speed of computers in the past few decades has led to a parallel increase in the number of applications. As with other advances in technology, as many problems are created as are solved. Although computers create a potential for solving many hitherto impossible problems the implementation is often much more difficult than might appear from cursory examination. However fast computer hardware becomes there will always be many problems whose computation takes a prohibitive amount of time. Many of these problems are of a "logical" nature. They arise in problems such as efficient computer circuit design, database retrieval systems, expert systems and artificial intelligence.

In order to explain the material in this chapter it is necessary to introduce some logical notation from the *propositional calculus* (also sometimes known as "sentential logic" or "boolean algebra").

We will use capital letters (sometimes suffixed) to represent *propositions* e.g. A, B, X_1, X_2, etc. These propositions may take two possible values, *true* (T) or *false* (F).

Such (atomic) propositions may be altered or combined by means of *connectives.*

e.g.	$\overline{A}$	means "not A"
	$A \vee B$	means "A or B (or both)"
	$A \cdot B$	means "A and B"

The truth of the resulting *compound propositions* depend on the truth or falsity of the component *atomic propositions* in a way prescribed by *truth tables.* Those connectives which we will use are defined by the truth table below.

A	B	$\overline{A}$	$A \vee B$	$A \cdot B$	$A \leftrightarrow B$	$A \to B$
F	F	T	F	F	T	T
F	T	T	T	F	F	T
T	F	F	T	F	F	F
T	T	F	T	T	T	T

" $\leftrightarrow$ " is referred to as "equivalent to" and " $\to$ " is referred to as "implies".

Atomic propositions, e.g. A, X_1, etc. or their negation, e.g. $\overline{A}$, $\overline{X}_1$, etc. are referred to as *literals*.

Any compound proposition will generally have many different *representations*, e.g. $(A \vee \overline{B}) \cdot C$ can be shown to always have the same truth value as $(A \cdot B \cdot C) \vee (\overline{B} \cdot C)$. In order to give comparable representations it is convenient to use one of two common *standard representations*. These are known as *disjunctive form* (DF) and *conjunctive form* (CF). Disjunctive form consists of a series of *conjunctions* (i.e. combined by " $\cdot$ ") of literals which are combined in a *disjunction* (i.e. by " $\vee$ "). For example, the following is a statement in DF:

$$(X_1 \cdot \overline{X}_2 \cdot X_3) \vee (\overline{X}_1 \cdot X_2 \cdot X_4) \vee (\overline{X}_3 \cdot X_5).$$

For convenience the brackets are often left out and the " $\cdot$ " connective assumed to be more binding than the " $\vee$ " connective. The component conjunctions, e.g. $X_1 \cdot \overline{X}_2 \cdot X_3$, are referred to as *conjunctive clauses*.

In contrast in CF a statement is written as a conjunction of disjunctive clauses, for example

$$(X_1 \vee X_2 \vee X_3) \cdot (\overline{X}_2 \vee X_3) \cdot (X_1 \vee \overline{X}_4 \vee \overline{X}_5)$$

is in CF where, e.g. $(X_1 \vee X_2 \vee X_3)$ is a disjunctive clause.

There are a number of standard equivalences between compound statements which can be used to convert statements into equivalent forms. The validity of these equivalences can readily be demonstrated by means of the truth table definitions above. Those equivalences which we will use are given below. We will use the symbol " $\equiv$ " to denote this (meta)equivalence, i.e. equivalence when referring *to* the system as opposed to sets of symbols *within* the system.

Reflexivity		$\overline{\overline{A}}$	$\equiv$	A
Symmetry	(i)	$A \vee B$	$\equiv$	$B \vee A$
	(ii)	$A \cdot B$	$\equiv$	$B \cdot A$
Distribution	(i)	$A \cdot (B \vee C)$	$\equiv$	$A \cdot B \vee A \cdot C$
	(ii)	$A \vee B \cdot C$	$\equiv$	$(A \vee B) \cdot (A \vee C)$
De Morgan's laws	(i)	$\overline{A \vee B}$	$\equiv$	$\overline{A} \cdot \overline{B}$
	(ii)	$\overline{A \cdot B}$	$\equiv$	$\overline{A} \vee \overline{B}$
Implies		$A \to B$	$\equiv$	$\overline{A} \vee B$
Equivalence		$A \leftrightarrow B$	$\equiv$	$(A \to B) \cdot (B \to A)$

Although we have confined ourselves to the propositional calculus it should be mentioned that some applications of computational logic require use of the *predicate calculus*. This allows the use of *predicates* whose truth values depend on the interpretation of *arguments*, e.g. $F(a)$ might represent the predicate "Father of" whose truth depended on the substitution of a value for the argument "a". Once such interpretations have been made (a process known as *instantiation*) the statements are reduced to those of the propositional calculus. Although we make partial use of the predicate calculus in Section 2.4 its use is mainly beyond the scope of this chapter.

Further details and a rigorous coverage of the propositional and predicate calculus can be found in, e.g. Mendelson [18].

We will demonstrate some typical problems which arise in computational logic by means of small examples. To start with we will use English sentences but later revert to the more compact abstract notation of using capital letters for sentences. These problems are given here for illustrative purposes. Methods of solution are given in Sections 3, 4 and 5.

Example 1.1 Valid deduction (taken from Mendelson [18])

Is the following deduction valid?

> *"If Jones did not meet Smith last night, then either Smith was the murderer or Jones is lying. If Smith was not the murderer, then Jones did not meet Smith last night and the murder took place after midnight. If the murder took place after midnight, then either Smith was the murderer or Jones is lying. Hence Smith was the murderer."*

It is convenient to rewrite this problem in symbolic form representing component sentences by capital letters as follows:

$$\begin{aligned}
X_1 &\equiv \text{“Jones did not meet Smith last night”},\\
X_2 &\equiv \text{“Smith was the murderer”},\\
X_3 &\equiv \text{“Jones is lying”},\\
X_4 &\equiv \text{“The murder took place after midnight”}.
\end{aligned}$$

The premises of the argument are

$$X_1 \rightarrow (X_2 \vee X_3), \tag{1.1}$$

$$\overline{X}_2 \rightarrow (X_1 \cdot X_4), \tag{1.2}$$

$$X_4 \rightarrow (X_2 \vee X_3). \tag{1.3}$$

The suggested conclusion is

$$X_2. \tag{1.4}$$

Whilst for a small example such as this the trying of all possible truth values for X_1, X_2, X_3 and X_4 by means of truth tables would test if the conclusion is valid such complete enumeration could be prohibitively demanding in space and time for larger problems. Methods of computational logic or integer programming (IP) could be used to better effect.

More complicated examples than these (often involving the predicate calcu-